2456 SIMULADOR D'UN PROCESSADOR AMB ADOBE FLASH PLAYER

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per
Carlos Ruiz Salvador
i dirigit per
Antonio José Velasco González

Bellaterra, 22 de juny de 2010

El sotasignat, Antonio José Velasco González

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en Carlos Ruiz Salvador.

I per tal que consti firma la present.

Signat: Antonio José Velasco González

Bellaterra, 22 de juny de 2010

TAULA DE CONTINGUTS

CAPÍTOL 1.- INTRODUCCIÓ

L'objectiu principal d'aquest projecte, per tal de matisar el títol del mateix, és crear un simulador d'un processador amb Adobe Flash CS3 per a reproduir-lo amb Adobe Flash Player. Aquest simulador té caràcter didàctic i es farà servir posteriorment per a propòsits educatius, fent-lo servir com a material en campus virtuals d'ensenyament.

Com a objectiu secundari està l'adaptació del mateix per a la reproducció a dispositius mòbils equipats amb Adobe Flash Lite Player. L'assoliment d'aquest objectiu permetrà l'execució en mobilitat de l'aplicació des de qualsevol dispositiu compatible que se l'hagi descarregat prèviament.

En referència a l'aplicació, val a dir que simularà el comportament de la màquina algorísmica anomenada FEMTOPROC, que és capaç d'interpretar 4 instruccions molt senzilles: ADD, AND, NOT i JZ (jump if zero). ADD i AND són instruccions que s'han d'executar en dos cicles o fases, necessitant NOT i JZ només una.

Les diferents instruccions que componen un programa seran emmagatzemades en una memòria de 64 posicions de 8 bits cadascuna i hi haurà un Banc de Registres amb 8 registres de 8 bits. A part, també hi haurà un registre auxiliar, anomenat B.

A part, s'hauran de mostrar també totes les senyals i altres elements que composen la màquina algorísmica.

La present memòria s'organitza en diferents capítols que analitzen els diferents aspectes a tenir presents per a desenvolupar i portar aquest projecte a bon terme. En primer lloc està l'estudi de viabilitat, seguit de l'anàlisi, el disseny i la implementació del projecte.

Els apartats han estat creats per a oferir una explicació clara i concisa de cada aspecte, sense ornaments innecessaris que no aporten res a l'explicació del projecte.

CAPÍTOL 2.- ESTUDI DE VIABILITAT

2.1.- Objectiu

El projecte a valorar al llarg d'aquest estudi de viabilitat és la realització d'una aplicació en Adobe Flash Lite® per a fins educatius. Aquesta està destinada a simular una màquina algorísmica que interpreti 4 instruccions diferents. Per tant, les instruccions i el funcionament han de ser senzills també de cara a l'usuari.

Aquesta màquina tindrà una memòria de 64 posicions de 8 bits cadascuna, on s'emmagatzemaran les instruccions i un banc de registre de 8 posicions de 8 bits, on es guardaran els diferents valor que es vagin obtenint durant l'execució.

Un dels requeriments més importants és que s'ha de trobar una tecnologia que permeti l'execució de l'aplicació en el major nombre de dispositius disponibles, havent estat suggerit l'ús de les directrius donades per Open Screen Project.

2.2.- Recursos

Per a la realització d'aquest projecte calen com a recursos hardware:

- PC amb els següents requisits mínims:
  - Processador Intel® Pentium® 4, Intel Centrino®, Intel Xeon®, Intel Core™ Duo o compatible.
  - 512MB of RAM.
  - 2.5GB d'espai lliure al disc dur.
  - Targeta gràfica amb resolució de 1.024x768 amb 16 bits de colors.
- Connexió a internet.
- Dispositiu mòbil per a proves. En aquest cas, una HTC P3300.

Per a la realització d'aquest projecte calen com a recursos software:

- Adobe® Flash® CS3 (versió demostrativa o educativa)
- Microsoft® Windows® XP SP2 o Windows Vista® 32 bits.
- Microsoft® Office Project versió demostrativa i Oracle® OpenOffice

2.3.- Anàlisi de costos

Tenint present que aquest projecte no és d'empresa i pressuposant que l'estudiant ja té equip informàtic basat en Windows® amb connexió a Internet, els costos de realització queden vénen especificats a la següent taula.

Recurs	Cost
Dispositiu mòbil	150,00 €
Versió Educativa Adobe® Flash®	174,00 €
Oracle® OpenOffice	0,00 €
Versió demo 60d Microsoft® Project	0,00 €
Total:	**324,00 €**

Taula 1: Cost de material alumne

En el cas d'haver estat un projecte de caire productiu a una empresa, els costos haurien quedat aproximadament de la següent manera:

Recurs	Cost
Dispositiu mòbil	150,00 €
Amortització dels PCs dels tècnics i l'analista (inclou llicència)	141,66 €
Amortització del MS Project de l'analista	30,00 €
Amortització del MS Office 2003 del tècnic programador	11,67 €
Amortització del Adobe Flash CS4 del tècnic programador	66,67 €
Total:	**400,00 €**

Taula 2: Cost de material empresa

Com es pot observar, els costos de material són molt semblants entre alumne i empresa. Això, que un principi por portar a engany induint a pensar que hagués suposat el mateix cost, no és del tot cert ja que en una empresa hi intervenen altres factors que incrementen el cost i que passo a esmentar a continuació.

Un punt a tenir present a l'empresa són els recursos humans o de personal que intervenen a totes les fases, els quans suposen un cost individual de:

Recurs	Cost
Analista	25€/h
Tècnic programador	20€/h
Tècnic de sistemes	17€/h
Tècnic de proves	15€/h

Taula 3: Recursos humans empresa

Això de cara al cost final del projecte suposaria un cost total afegir de:

Tasca	Personal Implicat	Duració	Cost(€)
Estudi de viabilitat	Analista	25h	625 €
Disseny de la interfície d'usuari	Tècnic programador	15h	300 €
Programació de l'aplicació	Tècnic programador	80h	1600 €
Proves locals	Tècnic de proves	20h	300 €
Proves de compatibilitat	Tècnic de proves	20h	300 €
Instal·lació l'empresa	Tècnic de sistemes	2h	34 €
Proves finals a l'empresa	Tècnic de proves	5h	75 €
Correccions finals a l'empresa	Tècnic programador	20h	400 €
Elaboració documentació final	Tècnic programador	35h	700 €
	Total:	**222h**	**4.334 €**

Taula 4: Cost de personal empresa

Com es pot apreciar, el cost de la realització per part d'un alumne (324 €) és mot inferior al cost de la realització per part d'una empresa (4.334 €).

El cost per part de l'alumne, en aquest cas concret, s'ha vist reduït enormement degut a que he pogut obtenir un dispositiu mòbil durant un temps aproximat de 2 mesos i he pogut fer servir la versió demostrativa del software Adobe® Flash® CS3 durant el període de temps necessari per a desenvolupar el projecte.

2.4.- Avaluació de riscos

Un dels principals riscos que es poden patir, inherent a qualsevol projecte, és el possible canvi de requisits del mateix. Aquest risc por endarrerir la planificació, cosa que s'ha de tenir present al model de desenvolupament finalment escollit. Una avantatge d'aquest projecte és que els requisits estan establerts des del principi i no han de patir cap mena de variació.

Un altre dels riscos que té aquest projecte és la possible incompatibilitat entre els diferents models de terminals, pantalles, mètodes d'entrada, sistemes operatius i navegadors d'Internet existents al mercat.

2.5.- Model de desenvolupament

Per a una bona planificació del projecte es farà servir un model de desenvolupament evolutiu incremental. En concret, un mètode que interactua amb el client, que en aquest cas és el director de projecte.

Això permet tenir una part del projecte que es pot presentar al client final, tenint-la pràcticament finalitzada, el qual decidirà si aquesta part la dóna com a correcta o s'ha de dur a terme aquelles modificacions que consideri oportunes. Si es donés aquest últim cas, s'hauria de tornar a treballar sobre aquella part que s'ha de modificar, però al mateix temps ja s'hauria començat a treballar en la següent.

Els passos claus en el procés són començar amb una implementació simple dels requeriments del sistema, i iterativament millorar la seqüència evolutiva de versions fins que el sistema complet estigui implementat. En cada iteració, es poden realitzar canvis en el disseny i es poden afegir noves funcionalitats i capacitats al sistema.

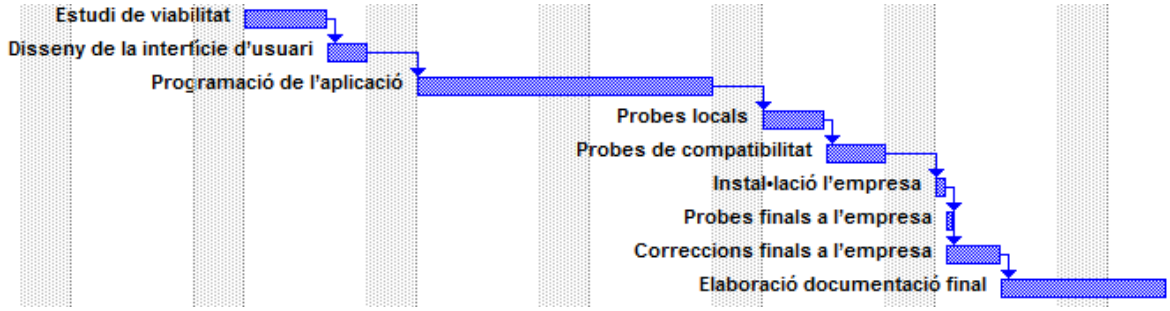
En resum, les característiques d'aquest model són:

- Permet estudiar i després millorar i ajustar el procés.
- No és aconsellable per a desenvolupaments llargs, ja que la interacció amb l'usuari pot incrementar encara més el temps.

2.6.- Planificació del projecte

A la taula destinada al cost de personal d'empresa (Taula 4) s'indiquen les fases per les quals el projecte haurà de passar, així com la durada aproximada de cadascuna de les tasques a realitzar.

Al diagrama de Gantt mostrat a continuació (Imatge 1) es pot observar la planificació del projecte en cas de que no s'hagués de modificar cap part del desenvolupament del projecte; cosa que implicaria un model de desenvolupament en cascada.



Imatge 1: Diagrama de Gantt

2.7.- Alternatives i possibilitats

Respecte al llenguatge de programació a emprar per aquest projecte, es van plantejar vàries possibilitats, totes força esteses al món de la telefonia mòbil:

- HTML5
- Oracle JME - J2ME
- Adobe Flash - Open Screen Project

2.7.1.- HTML5

Revisió 5 del llenguatge HTML, que estableix una nova sèrie d'elements i atributs que, en combinació amb JavaScript, CSS3 i Ogg, possibiliten crear aplicacions riques i millorar l'experiència web. Aquest model és recolzat àmpliament per Apple. Google també li dóna suport al seu sistema operatiu Android.

Aquest nou estàndard, que ara mateix està en fase inicial de desenvolupament i implantació al Web. Actualment, se li considera el gran rival de Flash, ja que permetria funcionalitats similars sense utilitzar *plugins* ni *runtimes* propietaris, com succeeix actualment. Un exemple clar és la visualització de videos a YouTube.

2.7.2.- Oracle JME - J2ME

Llenguatge de programació molt expandit dins el món de la telefonia mòbil, ja que permet la creació de tot tipus d'aplicacions com són els jocs que els usuaris de telefonia poden descarregar com a continguts Premium. Està més especialitzat en tasques de seguretat, accés als recursos del dispositiu i connectivitat (protocols d'Internet, Bluetooth, SMS...). Java és suportat a la pràctica totalitat de dispositius, incloent alguns que només tenen connectivitat WAP. És Opensource.

2.7.3.- Adobe Flash - Open Screen Project

Entorn de desenvolupament per a, principalment, crear aplicacions animades, multimèdia i interactives per la Web. El llenguatge de programació de Adobe Flash és ActionScript, que és orientat a objectes i permet expandir les possibilitats d'aquest software fins a crear complexes animacions.

Adobe Flash Lite és una versió "reduïda" de Adobe Flash creada especialment per a fer-se servir a dispositius mòbils que permet la reproducció d'arxius SWF molt similars als que es poden reproduir en un ordinador personal. Un avantatge és que aquell desenvolupador que hagi treballat anteriorment amb tecnologia Flash només haurà de tenir present a l'hora de programar aquelles limitacions inherents als dispositius mòbils (memòria, interfície d'entrada...).

Actualment el Open Screen Project, impulsat per Adobe, és una associació entre desenvolupadors de software i fabricants de hardware per a impulsar la creació d'aplicacions riques d'Internet (RIA, en anglès) mitjançant tecnologia Adobe i que siguin multiplataforma. Hi donen suport companyies com Nokia (Symbian), RIM (BlackBerry), Google (Android), Samsung, Antena 3... Actualment només un de tots els grans fabricants de dispositius mòbils no està en aquesta associació: Apple Inc., que aposta per la fusió HTML5+CSS3. Tot i que en un principi es podien crear aplicacions per iPhone amb Flash CS5, Apple ja no ho permet[1].

Després d'una primera valoració, valorant principalment el requeriment que s'esmenta més endavant d'arribar a la major part de dispositius possibles, en una primera fase HTML5 va quedar descartat ja que no són massa els dispositius mòbils que el suporten actualment, degut principalment al navegador que porten.

[1] Visiteu http://labs.adobe.com/technologies/packagerforiphone/ per a més informació

En una segona fase es van haver de comparar les dues tecnologies restants per valorar-les posteriorment. El resultat de la comparació, aspecte per aspecte, és:

	Java ME MSA	Flash Lite 2 ó 3
Gràfics i Multimèdia		
GIF		✓
JPEG	✓	✓
PNG	✓	✓
SVG	✓	✓
3D	✓	
FLA		✓
FLV		✓
3gpp	✓	✓
on2		✓
Sorensen		✓
MP3	✓	✓
video stream	✓	✓
audio stream	✓	✓
Accesibilitat al Dispositiu Local		
dades	✓	✓
I/O fitxers	✓	
calendari	✓	
contactes	✓	
càmera	✓	
micròfon	✓	
GPS	✓	
SIM	✓	
acceleròmetre		
marcació	✓	✓
Seguretat		
HTTPS	✓	✓
encriptació	✓	
Certificats	✓	

Taula 5: Comparativa entre JME i Flash Lite

Com es pot veure clarament, aquestes tecnologies estan enfocades, clarament, a necessitats de programació diferents. Degut a les característiques de l'aplicació que es vol desenvolupar,

Valorant les necessitats i requeriments del projecte, la meva experiència prèvia en Flash, i el repte de desenvolupar una versió per a mòbils tenint present les característiques "reduïdes" dels dispositius, vaig decidir finalment realitzar l'aplicació en Adobe Flash Lite versió 2.1 amb ActionScript 2.0. Això permetrà l'execució del programa a dispositius mòbils compatibles que porten més temps al mercat i a aquells d'última generació, ja que Adobe Flash Lite 3.0 reprodueix fitxers creats en la versió 2.0 ó 2.1.

2.8.- Conclusions

Aquest projecte té també com un dels principals avantatges que, un cop finalitzat, l'aplicació per si mateixa no requereix de cap tipus de manteniment ni d'actualització.

D'acord amb tot lo exposat als punts anteriors dins aquest apartat d'estudi de viabilitat, considero que el projecte és viable i, per tant, es pot tirar endavant.

CAPÍTOL 3.- ANÀLISI DEL PROJECTE

3.1.- Requeriments funcionals

Els requeriments funcionals són aquells que descriuen el comportament desitjat, del sistema, és a dir, especifica les sortides que s'han de produir a partir d'unes determinades entrades i les operacions necessàries per aconseguir-ho.

Els requeriments funcionals del projecte es descriuen a continuació:

Nom	Descripció
Consulta del manual de l'aplicació	A la pantalla inicial, l'usuari haurà de poder obtenir les instruccions de funcionament del programa.
Introduir el valor dels diferents registres	Com a pas previ a la introducció de dades, l'usuari haurà d'especificar els valors de cadascun dels registres que formen part del Banc de Registres.
Introduir les instruccions	L'usuari haurà de poder introduir les instruccions que desitgi, una a una.
Visualitzar l'execució de les instruccions	Per a cada instrucció introduïda, s'han de mostrar els valors de cada fase i, un cop finalitzat, passar a la següent instrucció. L'usuari ha de poder veure en tot moment el valor en memòria de la instrucció actual, l'anterior i la següent. Per a cada fase d'execució, s'han de poder distingir de la resta aquells busos, registres i valors que s'estan fent servir (llegint o escrivint) en aquell mateix instant.

3.2.- Requeriments no funcionals

Els requeriments no funcionals són aquells que descriuen restriccions que no afecten al funcionament ni al comportament del sistema. Aquestes restriccions afecten normalment al rendiment, al disseny, a la compatibilitat, etc.

3.2.1.- Rendiment

Com a requeriments de rendiment tenim els següents punts:

- El temps d'execució de cada fotograma no ha de ser superior a 750ms. Aquesta imposició ve donada per l'entorn de desenvolupament de Adobe Flash Lite, ja que si es sobrepassa aquest temps, surt un avís de que s'ha exhaurit el temps i l'execució s'atura.
- L'arxiu SWF generat que s'haurà de reproduir a cada dispositiu haurà de ser d'una grandària, en la mesura de lo possible, no superior a 250KB. Això permetrà una ràpida càrrega a la majoria de dispositius i s'evitarà l'aparició del *timeout* explicat al punt anterior.

3.2.2.- Disseny

Com a requeriments de disseny tenim els següents punts:

- La mida de l'escena ha de ser de 240x320 píxels.
- Durant l'execució, a la pantalla s'ha de visualitzar completament tot l'esquema del circuit de la màquina d'interpretació
- Les parts del circuit utilitzades durant el circuit, s'hauran de ressaltar en color vermell, per captar l'atenció de l'usuari.

3.2.3.- Plataforma i compatibilitat

Com a requeriments de plataforma i compatibilitat tenim els següents punts:

- S'ha de fer servir Flash Lite 2.0 o superior.
- S'ha de fer servir ActionScript 2.0 o superior, en concordança amb l'elecció triada al punt anterior.

3.3.- Model de casos d'ús

Els casos d'ús representen els requeriments funcionals del sistema i, per tant, especifiquen els comportament desitjat dels mateixos.

A continuació es mostren els diferents casos d'ús indicant per cadascun d'ells l'actor, els fluxos i les diferents condicions, si existeixen.

Nom	**Consulta del manual de l'aplicació**
Descripció	Permet consultar el manual d'usuari de l'aplicació.
Actors	Usuari
Precondició	---
Flux principal	1. Per accedir al manual, l'usuari (actor) haurà de seleccionar el botó "Instruccions" a la pantalla inicial. 2. L'actor podrà canviar de pàgina amb els botons destinats a tal efecte en cada pantalla del manual.
Flux alternatiu	---
Postcondició	Un cop arribat al final del manual, l'actor tornarà a la pantalla inicial.

Nom	**Introduir el valor dels diferents registres**
Descripció	Permet introduir els valor dels registres.
Actors	Usuari
Precondició	L'actor ha d'haver premut el botó "INICIAR", a la pantalla inicial de l'aplicació.
Flux principal	1. Per a cada registre amb valors modificables (R2 a R7), l'actor haurà d'introduir un valor en binari de 8 dígits. 2. Un cop introduïts, l'actor premerà "Següent". 3. L'aplicació comprovarà la validesa de les dades introduïdes. 4. Si són correctes, l'aplicació passarà a la següent fase.
Flux alternatiu	2. Si prem "Cancelar", tornarà a la pantalla inicial. 4. Si no són correctes, apareixerà un missatge indicant que hi ha un error. 5. L'actor romandrà a la mateixa pantalla, per a modificar aquells valors incorrectes.
Postcondició	---

Nom	Introduir les instruccions
Descripció	Permet introduir les instruccions del programa a memòria.
Actors	Usuari
Precondició	L'actor ha d'haver introduït correctament el valor dels registres a la pantalla anterior.
Flux principal	1. Per a cada instrucció que l'actor vulgui introduir al programa, haurà de seleccionar el tipus i introduir els operands corresponents. 2. Un cop introduïts, l'actor premerà "Següent", si vol introduir una altra, o "OK" si vol desar la instrucció actual i passar a la fase d'execució. 3. L'aplicació comprovarà la validesa de les dades introduïdes. 4. Si són correctes, l'aplicació passarà a la següent fase, segons lo seleccionat al punt 2 anterior.
Flux alternatiu	2. Si prem "Cancelar", tornarà a la pantalla inicial. 4. Si no són correctes, apareixerà un missatge indicant que hi ha un error. 5. L'actor romandrà a la mateixa pantalla, per a modificar aquells valors incorrectes.
Postcondició	---

Nom	**Visualitzar l'execució de les instruccions**
Descripció	Per a cada instrucció introduïda, s'han de mostrar els valors de cada fase i, un cop finalitzat, passar a la següent instrucció. L'usuari ha de poder veure en tot moment el valor en memòria de la instrucció actual, l'anterior i la següent. Per a cada fase d'execució, s'han de poder distingir de la resta aquells busos, registres i valors que s'estan fent servir (llegint o escrivint) en aquell mateix instant.
Actors	Usuari
Precondició	L'actor ha d'haver introduït correctament les instruccions prèviament.
Flux principal	1. Per a cada fase d'execució de cada instrucció o a cada instrucció, segons correspongui, s'han de mostrar els valors actuals a la memòria, registres i banc de registres. 2. Un cop mostrats, l'aplicació quedarà en espera fins que l'actor premi "Següent Fase" o "Següent Instrucció", segons el cas. 3. Si n'hi més a continuació, l'aplicació passarà a mostrar l'execució de la següent fase o instrucció.
Flux alternatiu	3. Si no hi ha cap més instrucció, l'aplicació passarà a l'estat final d'execució, indicant-ho a la pantalla i mostrant els valors finals. 4. L'actor, al prémer "Fi del Programa", tornarà a la pantalla inicial.
Postcondició	---

CAPÍTOL 4.- DISSENY DE L'APLICACIÓ

4.1.- Disseny de les dades

El disseny de les dades permet definir els processos i característiques de les dades de l'aplicació.

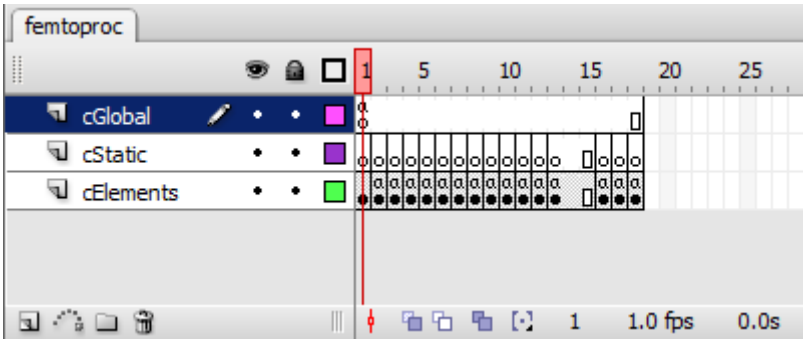
En el cas concret d'aquesta aplicació a desenvolupar, no és necessari cap tipus de base de dades, ja que les dades només es faran servir durant el temps d'execució de la mateixa. No cal desar-les ni tampoc cal accedir-hi posteriorment per a la seva consulta.

A continuació s'exposen les característiques de les funcions globals i de les principals variables del programa que allotgen les dades:

```
_global.validar_regs = function(valor:String):Boolean
```
funció que valida si els valors introduïts als registres són correctes
```
_global.bin_a_hex = function(valor:String):String
```
funció que converteix les cadenes de text de binari a hexadecimal
```
var num_ins = 0;
```
variable que indica la instrucció actual (PC)
```
var array_ins:Array = Array();
```
variable que conté totes les instruccions del programa (memòria)
```
var Z = 0;
```
variable que indica si la operació anterior ha donat 0 o no.
```
var RegistreB:String = "00000000";
```
variable on es guarda el valor del Registre B i que s'inicialitza a 00000000
```
var Registres:Array = Array();
```
variable on es guarden els valors dels registres del Banc de Registres
```
var format_vermell:TextFormat = new TextFormat();
format_vermell.color = 0xFF0000;
```
variable de format de text de Flash que serveix per canviar el color del text de l'element on s'apliqui

4.2.- Disseny arquitectònic

L'entorn Adobe Flash té unes característiques especials que s'han de tenir presents i que s'anomenen a continuació:

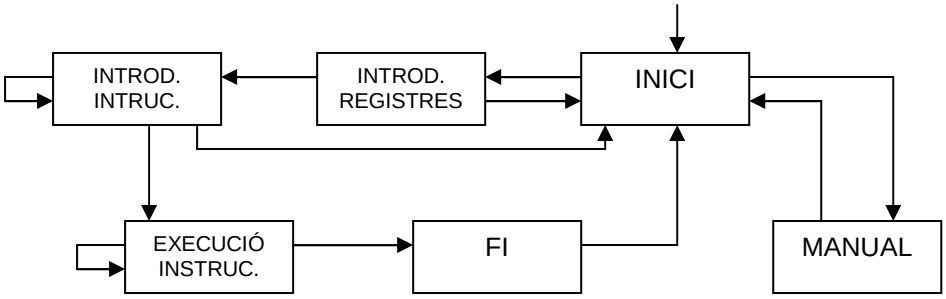


Imatge 2: Línia de temps de Adobe Flash CS3

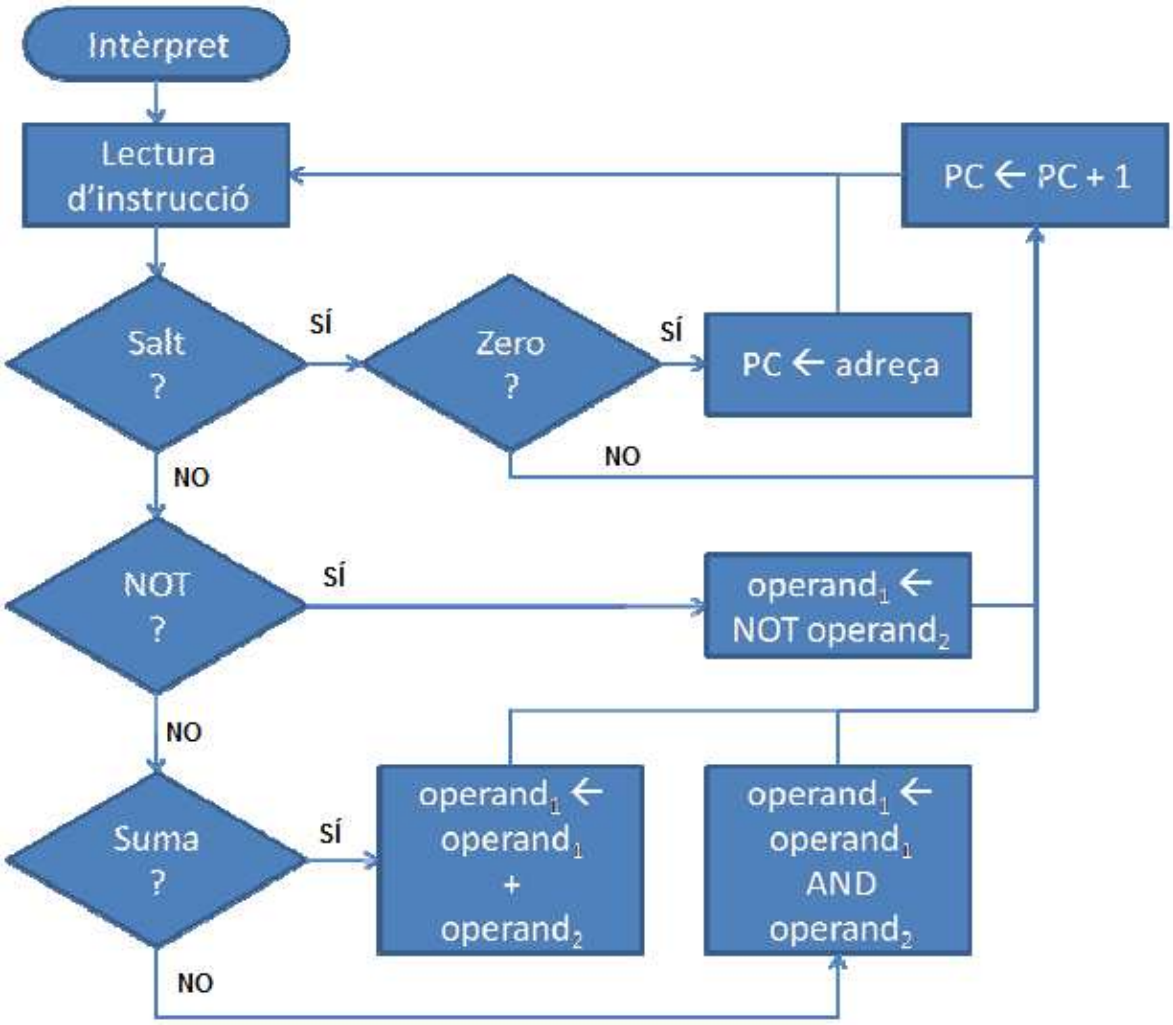
- Número de fotograma: indicant a la part superior el número que correspon, serveix per a poder establir un cert ordre dins de la pel·lícula (nom que rep el contingut de l'arxiu amb extensió .FLA) per fer salts o per establir la durada de cada escena.
- Capes: normalment cada capa de la pel·lícula conté un objecte (i una animació o no del mateix). L'ordre de reproducció de les capes es pot configurar segons les preferències. Per a explicar-ho de forma més entenedora, comentaré les 3 capes d'aquest exemple:
    - cGlobal: capa on es guarden les variables i funcions globals, comunes a tota la pel·lícula. D'aquí la seva extensió des del principi al final de la pel·lícula.
    - cStatic: on s'emmagatzemen els elements fixos de cada fotograma.
    - cElements: on per cada fotograma hi tenim aquells elements dinàmics o que poden variar, afectats pel codi ActionScript. A cada fotograma hi apareix una "a" que indica que hi ha una acció.
- Velocitat: a la part inferior dreta de la imatge es pot veure la velocitat de l'animació, mostrat en fotogrames per segon (fps).
- Temps: a la dreta de la part on s'indica la velocitat, es mostra el temps al qual es mostraria el frame actual seleccionat (indicat per la línia vermella), en cas de que l'animació fos totalment lineal.

4.3.- Disseny procedimental

Aquest tipus de disseny ha d'especificar els detalls dels procediments. Per a això, s'ha revisat la informació provinent dels requeriments especificats en l'apartat "Anàlisi del Projecte" d'aquesta memòria. La imatge mostra l'esquema de flux del sistema.



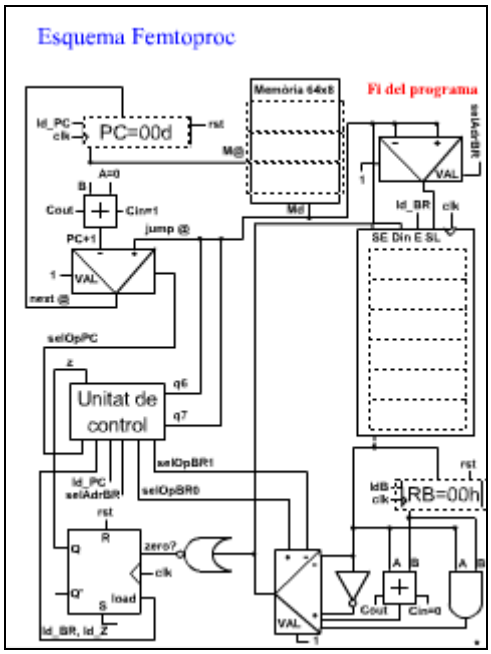
Imatge 3: Diagrama de flux del sistema



Imatge 4: Diagrama de flux de la màquina d'interpretació de les instruccions

4.4.- Disseny d'interfícies

Pel disseny s'han seguint les pautes definides als requeriments no funcionals. S'ha buscat senzillesa d'utilització i claredat a l'hora de mostrar les dades per pantalla, augmentant en la mesura de lo possible la mida de les fonts dels camps dinàmics per a una millor visualització.



Imatge 5: Disseny final per a la pantalla de 240x320 píxels

CAPÍTOL 5.- IMPLEMENTACIÓ

Per a la realització d'aquest projecte he fet servir les eines proporcionades per la suite Adobe Flash CS3[2].

Al final, com arxius resultants s'han obtingut:

- femtoproc.fla → arxiu on es programa tota l'animació, amb el codi font.
- femtoproc.swf → arxiu resultant de compilar i publicar l'animació anterior.
- i → arxius instal·ladors de Flash Lite Player.

5.1.- Adobe Device Central CS3

Aquest és un component integrat de Adobe Flash CS3 que permet visualitzar els arxius creats com si estiguessis sent executats en un dispositiu mòbil, amb la mateixa interfície o una de molt semblant. Aquest complement inclou una llibreria amb diversos terminals disponibles.

5.2.- Configuració de la Publicació dins Adobe Flash CS3

Aquest apartat s'haurà de configurar lo següent:

- Versió: Flash Lite 2.1
- Ordre de capa: de dalt a baix
- Versió de ActionScript: ActionScript 2.0

5.3.- Biblioteca, Panell d'Eines, Propietats, Paràmetres i Accions

Per a la creació dels diferents elements de l'animació he hagut de fer servir, principalment, totes les possibilitats d'aquestes barres d'eines i pestanyes.

5.4.- ActionScript

Aquest llenguatge de programació propi de Adobe ha estat el que realment he hagut de fer servir per a crear l'animació i mostrar tots els valors i elements amb un comportament dinàmic.

[2] Veure la bibliografia completa feta servir al capítol 8.

CAPÍTOL 6.- PROVES REALITZADES

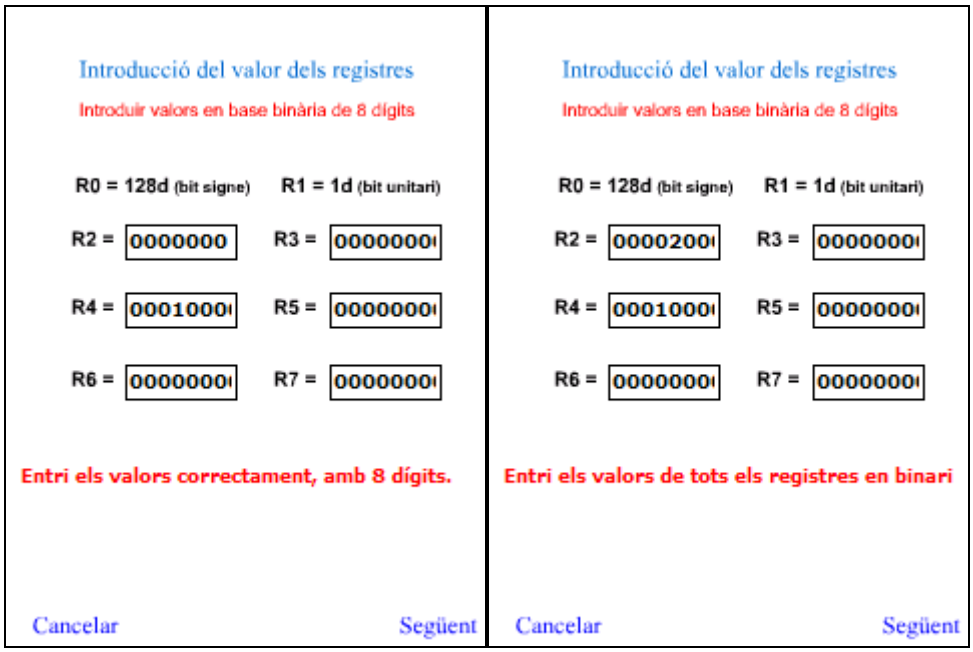
Per a la fase de test de l'aplicació, he realitzat les proves que s'especifiquen a continuació:

6.1.- Prova 1 – Manual

Es tracta de comprovar si quan s'accedeix a les instruccions de l'aplicació, al final aquesta torna al començament. Resultat: OK.

6.2.- Prova 2 – Valors Registres

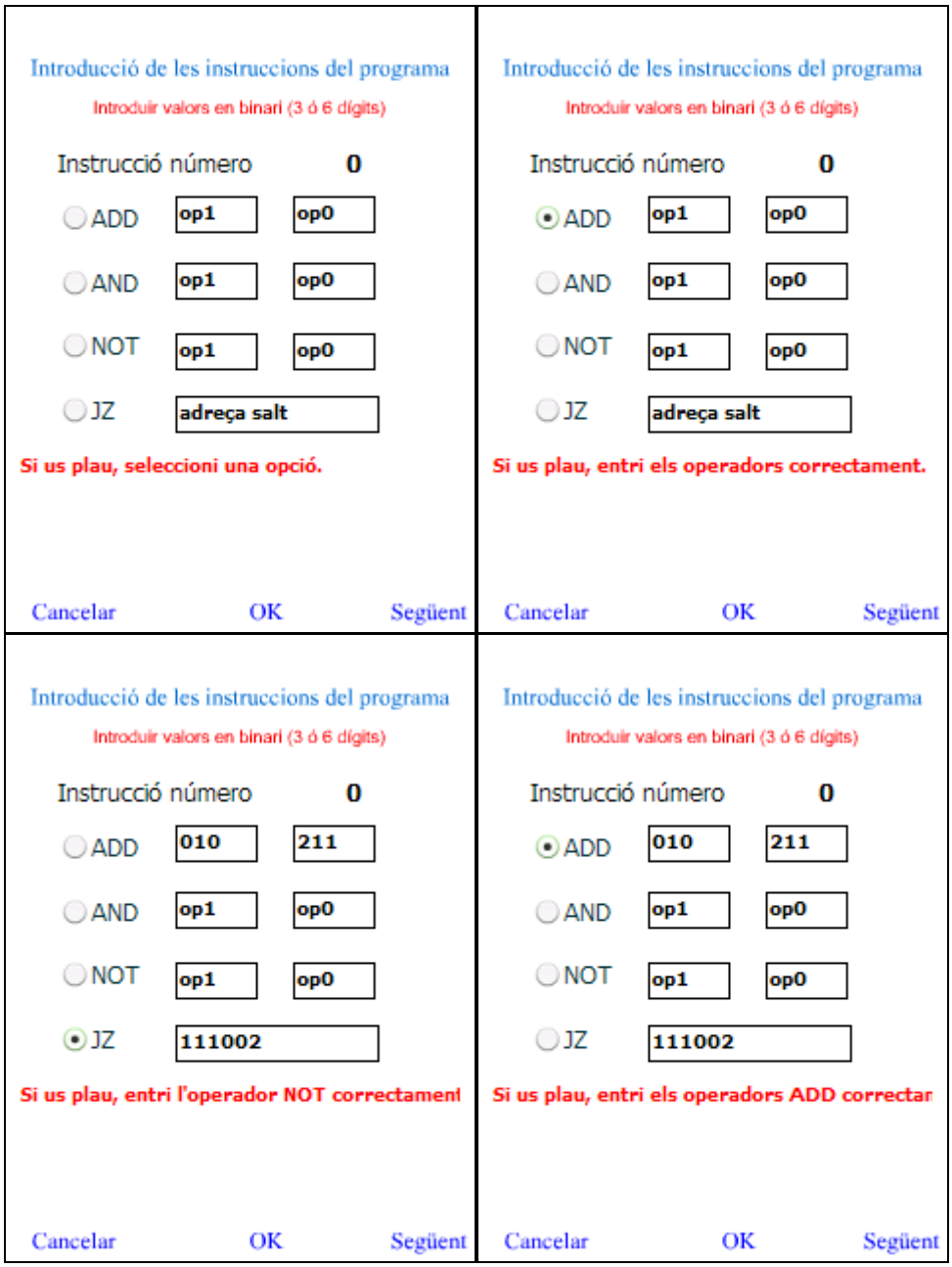
Es tracta de comprovar si quan s'introdueixen els valors correctament l'aplicació continua amb el següent procediment; i si quan s'introdueixen incorrectament surt un missatge d'error i s'informa a l'usuari perquè ho corregeixi. Resultat: OK.



Imatge 6: Resultats Prova 2

6.3.- Prova 3 – Valors Operands Instruccions

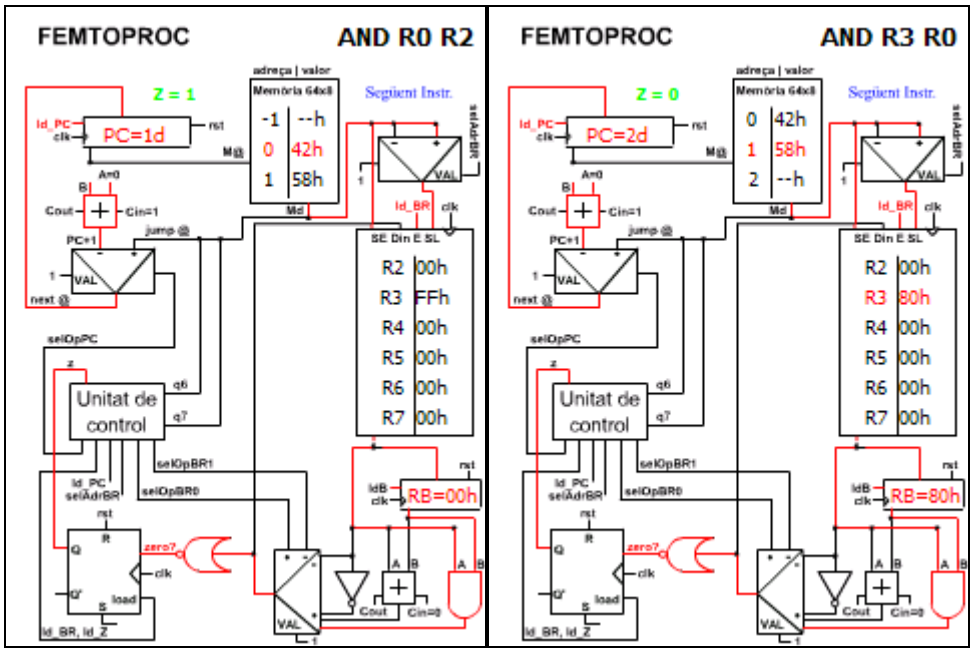
Es tracta de comprovar si quan s'introdueixen els valors dels operands correctament l'aplicació continua amb el següent procediment; i si quan s'introdueixen incorrectament surt un missatge d'error i s'informa a l'usuari perquè ho corregeixi. Resultat: OK.



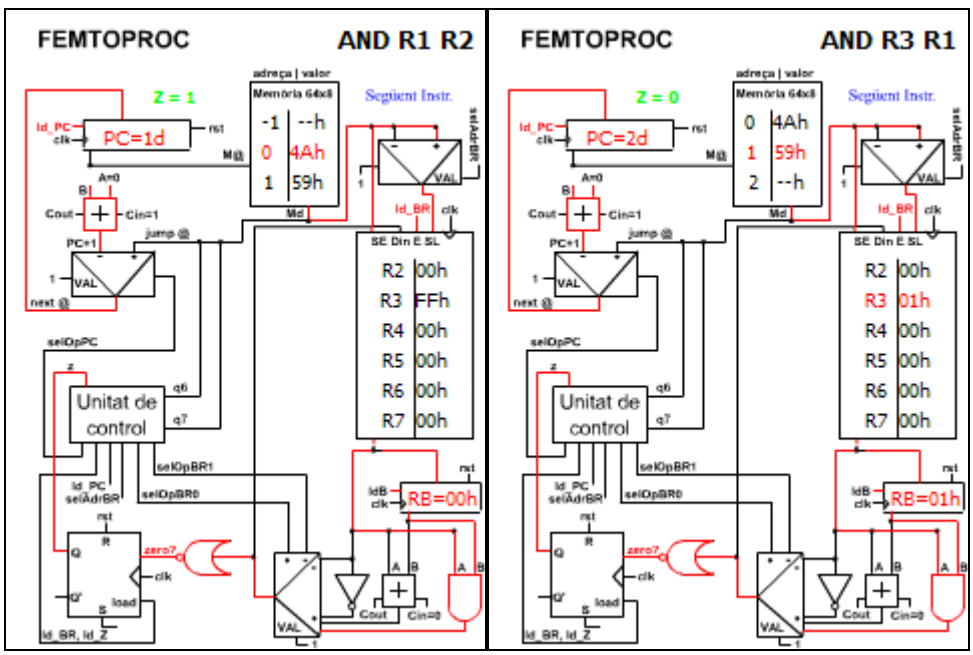
Imatge 7: Resultats Prova 3

6.4.- Prova 4 – Prova de no escriptura a R0 i R1

Es tracta de comprovar si quan s'introdueixen els valors dels registres R0 i R1 com a operands de destí del resultat, aquestos no queden modificats després d'executar la instrucció corresponent. Això quedarà reflectit si en un programa de 2 instruccions seguides el resultat és aquell que s'esperava. Resultat: OK.



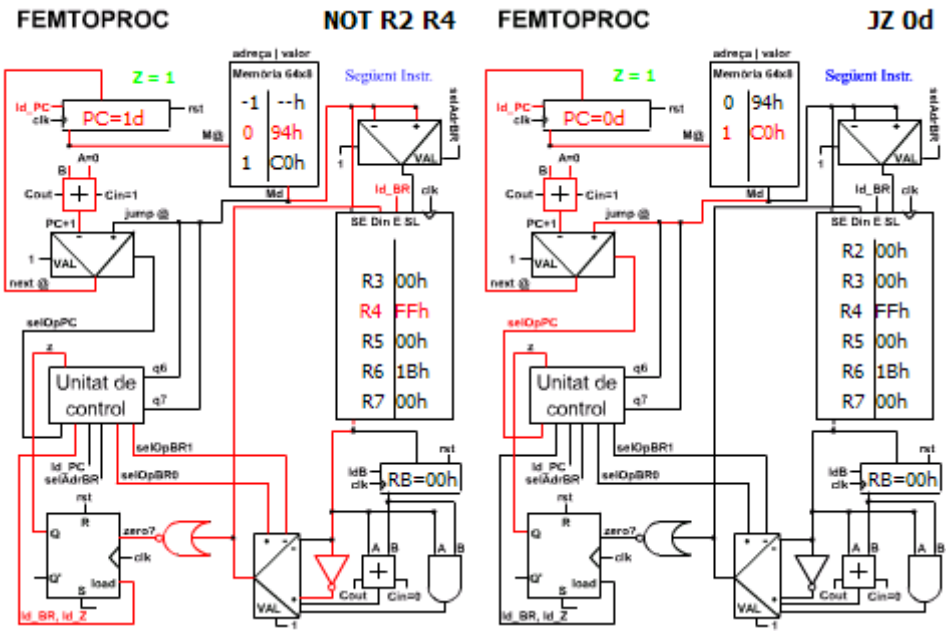
Imatge 8: Resultats Prova 4 – R0 = R3 = 80h



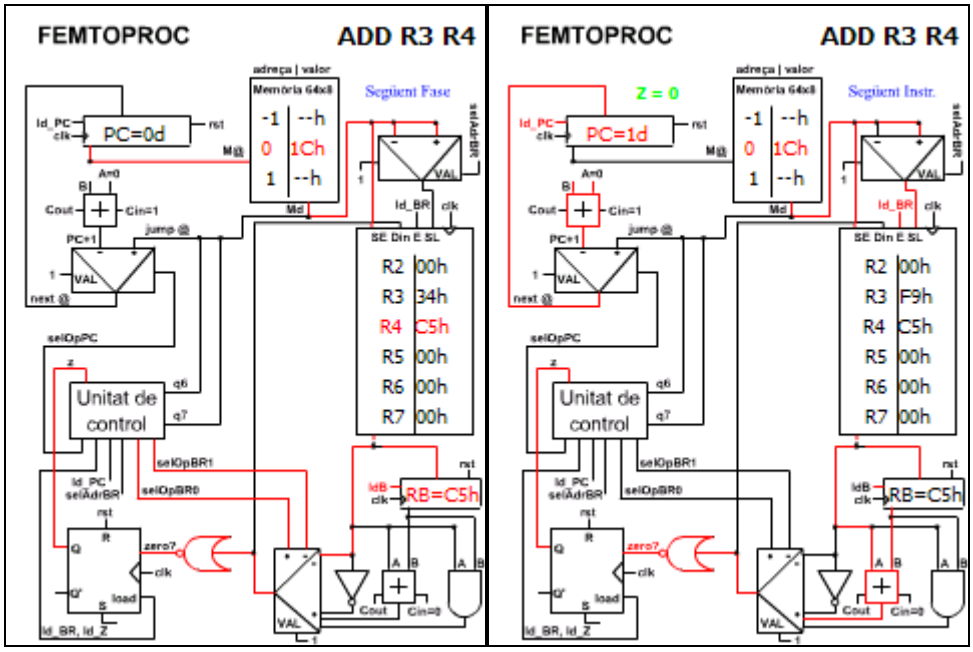
Imatge 9: Resultats Prova 4 – R1 = R3 = 01h

6.5.- Prova 5 – Marcació elements correctes per cada fase

Es tracta de comprovar si, per cada instrucció o fase, l'aplicació assenyala correctament els busos i senyals que estan actius en aquell moment, per així mostrar clarament a l'usuari el funcionament del processador. Resultat: OK.



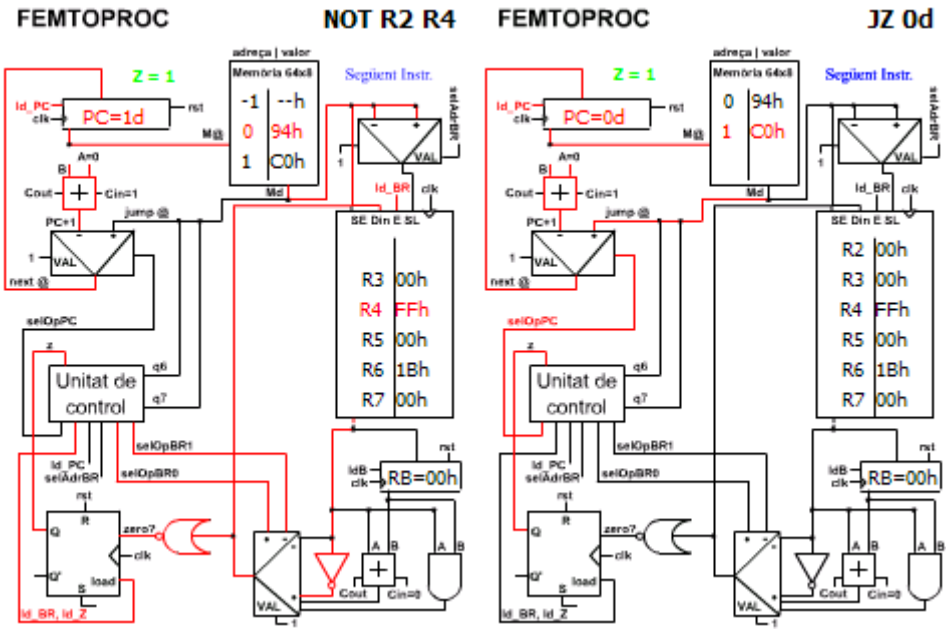
Imatge 10: Resultats Prova 5 – NOT i JZ



Imatge 11: Resultats Prova 5 – ADD

6.6.- Prova 6 – Comprovar si Z=0 provoca salt a nova adreça

Es tracta de comprovar si, quan la instrucció anterior a una JZ dóna com a resultat Z=0, la instrucció JZ salta a l'adreça especificada. Resultat: OK.



Imatge 12: Resultats Prova 6 – NOT amb Z=0 i JZ amb PC=0d

CAPÍTOL 7.- CONCLUSIONS

7.1.- Conclusions

Per tot lo comentat al llarg d'aquesta memòria, puc concloure que el projecte ha resultat ser definitivament viable i que, a més a més, s'han complit els objectius i requeriments especificats a les fases d'anàlisi i de disseny.

Com a valoració personal, aquest projecte ha suposat un nou repte a la meva experiència personal ja que, tot i haver realitzat algunes webs en Flash, aquesta és la primera vegada que he realitzat una aplicació per a dispositius mòbils.

Aquesta primera immersió al món dels dispositius mòbils m'ha servit per aprofundir molt en aquest món, conèixer altres tecnologies i, principalment, veure cap a on tendeix el món Web actual, molt pendent de HTML i de les batalles entre fabricants. Tot això ve donat perquè avui en dia aquest tipus de dispositius es consideren una gran font d'ingressos per a les companyies.

7.2.- Ampliacions futures

Malgrat haver aconseguit acomplir, com ja s'ha comentat anteriorment, els requeriments i objectius, aquest projecte es podria ampliar i millorar en els següents aspectes:

- Millorar i preparar l'aplicació i el disseny gràfic de forma que hi hagi més mides de pantalla compatibles, sense perdre funcionalitat ni informació.
- Optimitzar i conèixer a fons tots els possibles models de telèfon del mercat per saber les seves CPUs, RAM, etc. i així obtenir una compatibilitat amb dispositius molt alta.

CAPÍTOL 8.- BIBLIOGRAFIA

1. Ribas, Lluís. *Fonaments de Computadors. Estructura bàsica d'un computador*. Universitat Oberta de Catalunya, Barcelona, 2010 (en elaboració). Pàgines 18-25.

2. Adobe Systems Inc & Partners. *Open Screen Project.* San José, California, EE.UU. 2009. Disponible al Web: http://www.openscreenproject.org/

3. Adobe Systems Inc. *Adobe Mobile & Devices Developers.* San José, California, EE.UU. 2010. Disponible al Web: http://www.adobe.com/devnet/devices/index.html?view=home

4. Nokia Corporation. *Open Screen Project Fund by Forum Nokia.* Espoo, Finlàndia. 2009. Disponible al Web: http://openscreen.forum.nokia.com/

5. Oracle. *Java ME Developers Network*. Disponible al Web: http://java.sun.com/javame/index.jsp

6. Oracle. *J2ME Mobile Information Device Profile.*. Disponible al Web: http://java.sun.com/products/midp/overview.html

7. Hickson, Ian i Hyatt, David. *HTML5 W3C Working Draft*. 04/03/2010. Disponible al Web: http://www.w3.org/TR/html5/

8. "The Man in Blue". *"HTML5" versus Flash: Animation Benchmarking.* 22/03/2010. Disponible al Web: http://www.themaninblue.com/writing/perspective/2010/03/22/

9. Rush, Keldon. *GETTING STARTED WITH ADOBE FLASH® LITE™*. San José, California, EE.UU. 2007. Disponible al Web: http://www.adobe.com/devnet/devices/articles/getting_started_flashlite_b.pdf

10. Adobe Systems Inc*. Documentación de Flash CS3.* San José, California, EE.UU. 2007. Disponible al Web: http://livedocs.adobe.com/flash/9.0_es/main/wwhelp/wwhimpl/js/html/wwhelp.htm

11. Adobe Systems Inc. *Utilización de Flash® 9.0 para Windows® y Macintosh*. San José, California, EE.UU. 2007. Disponible al Web: http://livedocs.adobe.com/flash/9.0_es/UsingFlash/

12. Adobe Systems Inc. *INTRODUCCIÓN A ACTIONSCRIPT™ EN FLASH® LITE™ 2.x.* San José, California, EE.UU. 2007. Disponible al Web: http://livedocs.adobe.com/flash/9.0_es/main/flashlite2_as_intro.pdf

13. Adobe Systems Inc. *REFERENCIA DEL LENGUAJE ACTIONSCRIPT™ DE FLASH® LITE™ 2.X.* San José, California, EE.UU. 2007. Disponible al Web: http://livedocs.adobe.com/flash/9.0_es/main/flashlite2_as_reference.pdf

14. Adobe Systems Inc. *ADOBE® FLASH® LITE™ 2.x y 3.x Referencia del lenguaje Adobe® ActionScript®.* San José, California, EE.UU. 2008. Disponible al Web: http://help.adobe.com/es_ES/FlashLite/2.0_FlashLiteAPIReference2/flashlite_2.x_3.x_aslr.pdf

ANNEXE

A continuació es mostren les característiques principals del codi de l'aplicació.

1.- Codi i Funcions a la capa cGlobal

```
stop();
var num_ins = 0;
//inicialització array Memòria
var array_ins:Array = Array();
for (var i:Number = 0; i<64; i++) {
	array_ins[i] = "";
}
var ult_ins = "";
var msg_ins = "";
var ins_act = 0;
var Z = 0;//Z==0 NO SALT - Z==1 SALT
//variables que ens permitiran distingir les instruccions
var str1:String = "00000000";
var str2:String = "01000000";
var str3:String = "10000000";
var str4:String = "11000000";
var next_ins:String = "";
var RegistreB:String = "00000000";
//inicialitzar array del Banc de Registres
var Registres:Array = Array();
Registres[0] = "10000000";
Registres[1] = "00000001";
Registres[2] = "00000000";
Registres[3] = "00000000";
Registres[4] = "00000000";
Registres[5] = "00000000";
Registres[6] = "00000000";
Registres[7] = "00000000";

var format_vermell:TextFormat = new TextFormat();
format_vermell.color = 0xFF0000;


_global.validar_regs = function(valor:String):Boolean  {

	var validat:Boolean = true;
	var longitud:Number = valor.length;

	//Bucle per fer la comprovació bit a bit
	for (var i:Number = 0; i<longitud; i++) {
		if ((valor.substr(i, 1) == "0") || (valor.substr(i, 1) == "1")) {
```

```
                        //no fem res, ja que és OK.
                } else {
                        validat = false;
                }
        }
        return validat;//si TRUE, registre OK; si FALSE, té dígit erroni
};

_global.bin_a_hex = function(valor:String):String  {

        var part1:String = valor.substr(0, 4);
        var part2:String = valor.substr(4, 4);

        var part1hex:String = "";
        var part2hex:String = "";

        if (part1 == "0000") {
                part1hex = "0";
        } else if (part1 == "0001") {
                part1hex = "1";
        } else if (part1 == "0010") {
                part1hex = "2";
        } else if (part1 == "0011") {
                part1hex = "3";
        } else if (part1 == "0100") {
                part1hex = "4";
        } else if (part1 == "0101") {
                part1hex = "5";
        } else if (part1 == "0110") {
                part1hex = "6";
        } else if (part1 == "0111") {
                part1hex = "7";
        } else if (part1 == "1000") {
                part1hex = "8";
        } else if (part1 == "1001") {
                part1hex = "9";
        } else if (part1 == "1010") {
                part1hex = "A";
        } else if (part1 == "1011") {
                part1hex = "B";
        } else if (part1 == "1100") {
                part1hex = "C";
        } else if (part1 == "1101") {
                part1hex = "D";
        } else if (part1 == "1110") {
                part1hex = "E";
        } else if (part1 == "1111") {
                part1hex = "F";
        } else {
                part1hex = "-";
        }
```

```
        if (part2 == "0000") {
                part2hex = "0";
        } else if (part2 == "0001") {
                part2hex = "1";
        } else if (part2 == "0010") {
                part2hex = "2";
        } else if (part2 == "0011") {
                part2hex = "3";
        } else if (part2 == "0100") {
                part2hex = "4";
        } else if (part2 == "0101") {
                part2hex = "5";
        } else if (part2 == "0110") {
                part2hex = "6";
        } else if (part2 == "0111") {
                part2hex = "7";
        } else if (part2 == "1000") {
                part2hex = "8";
        } else if (part2 == "1001") {
                part2hex = "9";
        } else if (part2 == "1010") {
                part2hex = "A";
        } else if (part2 == "1011") {
                part2hex = "B";
        } else if (part2 == "1100") {
                part2hex = "C";
        } else if (part2 == "1101") {
                part2hex = "D";
        } else if (part2 == "1110") {
                part2hex = "E";
        } else if (part2 == "1111") {
                part2hex = "F";
        } else {
                part2hex = "-";
        }
        var retornar = part1hex+part2hex+"h";
        return retornar;//si TRUE, registre OK; si FALSE, té dígit erroni
};
```

2.- Codi a la capa cDinamic

Relatives a la validació de dades dels Registres

```
on (release) {
	var continua:Boolean = true;
	if ((r2.text.length == 8) && (r3.text.length == 8) && (r4.text.length == 8)
&& (r5.text.length == 8) && (r6.text.length == 8) && (r7.text.length == 8)) {
		for (var i:Number = 2; i<8; i++) {
			var Registre_Entrada:String;
			if (i == 2) {
				Registre_Entrada = r2.text;
			} else if (i == 3) {
				Registre_Entrada = r3.text;
			} else if (i == 4) {
				Registre_Entrada = r4.text;
			} else if (i == 5) {
				Registre_Entrada = r5.text;
			} else if (i == 6) {
				Registre_Entrada = r6.text;
			} else {
				Registre_Entrada = r7.text;
			}
			var es_valid:Boolean = _global.validar_regs(Registre_Entrada);
			if (es_valid == true) {
				Registres[i] = Registre_Entrada;
			} else {
				continua = false;
			}
		}
		if (continua == false) {
			msg_instr.text = "Entri els valors de tots els registres en
binari.";
		} else {
			gotoAndPlay(3);
		}
	} else {
		msg_instr.text = "Entri els valors correctament, amb 8 dígits.";
	}
}
```

Relatives a la validació de les Instruccions introduïdes

```
on (release)
{
    if(radioGroup.selection.label == 'ADD') {
                if((_global.validar_regs(addop1.text))  &&
(_global.validar_regs(addop0.text))) {
                        msg_inst.text = "Ha introduït correctament: 00" + addop1.text
+ addop0.text;
                        array_ins[num_ins] = "00" + addop1.text + addop0.text;
                        gotoAndPlay(5);
                }
                else
                {
                        msg_inst.text = "Si us plau, entri els operadors ADD
correctament.";
                }
        }
    else if (radioGroup.selection.label == 'AND'){
                if((_global.validar_regs(andop1.text))  &&
(_global.validar_regs(andop0.text))) {
                        msg_inst.text = "Ha introduït correctament: 01" + andop1.text
+ andop0.text;
                        array_ins[num_ins] = "01" + andop1.text + andop0.text;
                        gotoAndPlay(5);
                }
                else
                {
                        msg_inst.text = "Si us plau, entri els operadors AND
correctament.";
                }
        }
    else if (radioGroup.selection.label == 'NOT'){
                if((_global.validar_regs(notop1.text))  &&
(_global.validar_regs(notop0.text))) {
                        msg_inst.text = "Ha introduït correctament: 10" + notop1.text
+ notop0.text;
                        array_ins[num_ins] = "10" + notop1.text + notop0.text;
                        gotoAndPlay(5);
                }
                else
                {
                        msg_inst.text = "Si us plau, entri els operadors NOT
correctament.";
                }
        }
    else if (radioGroup.selection.label == 'JZ'){
                if(_global.validar_regs(jzop0.text)) {
                        msg_inst.text = "Ha introduït correctament: 11" + jzop0.text;
                        array_ins[num_ins] = "11" + jzop0.text;
                        gotoAndPlay(5);
                }
                else
                {
                        msg_inst.text = "Si us plau, entri l'operador NOT
correctament.";
                }
        }
    else {
        msg_inst.text = "Si us plau, seleccioni una opció.";
    }
}
```

Relatives a la selecció de la instrucció a executar

```
//stop();
next_ins = array_ins[ins_act];
if (_global.bin_a_hex(next_ins) == "--h") {
	gotoAndStop(16);
} else {
	if (next_ins>=str4) {
		//trace("Operació 11");
		gotoAndPlay(6);
	} else if (next_ins>=str3) {
		//trace("Operació 10");
		gotoAndPlay(7);
	} else if (next_ins>=str2) {
		//trace("Operació 01");
		gotoAndPlay(8);
	} else if (next_ins>=str1) {
		//trace("Operació 00");
		gotoAndPlay(10);
	}
}
```

Relatives a l'execució de la instrucció (1a fase)

```
stop();
var andop1reg:Number;
var andop2reg:Number;
var andop1:String = next_ins.substr(2, 3);
var andop2:String = next_ins.substr(5, 3);
if (andop1.substr(0, 1) == "0") {
	var op1_temp_str:String = "1"+andop1;
	var op1_temp_dec:Number = parseInt(op1_temp_str, 2);
	andop1reg = op1_temp_dec-8;
} else {
	andop1reg = parseInt(andop1, 2);
}
if (andop2.substr(0, 1) == "0") {
	var op2_temp_str:String = "1"+andop2;
	var op2_temp_dec:Number = parseInt(op2_temp_str, 2);
	andop2reg = op2_temp_dec-8;
} else {
	andop2reg = parseInt(andop2, 2);
}

and_ops.text = "AND R"+andop1reg+" "+"R"+andop2reg;

RegistreB = Registres[andop2reg];

pc_ins_and1.text = "PC="+ins_act+"d";
rb_ins_and1.text = "RB="+_global.bin_a_hex(RegistreB);
r2_ins_and1.text = "R2  "+_global.bin_a_hex(Registres[2]);
r3_ins_and1.text = "R3  "+_global.bin_a_hex(Registres[3]);
r4_ins_and1.text = "R4  "+_global.bin_a_hex(Registres[4]);
r5_ins_and1.text = "R5  "+_global.bin_a_hex(Registres[5]);
r6_ins_and1.text = "R6  "+_global.bin_a_hex(Registres[6]);
r7_ins_and1.text = "R7  "+_global.bin_a_hex(Registres[7]);
mem0_ins_and1.text = (ins_act-1) +"   "+_global.bin_a_hex(array_ins[ins_act-1]);
mem1_ins_and1.text = (ins_act) +"   "+_global.bin_a_hex(array_ins[ins_act]);
mem2_ins_and1.text = (ins_act+1) +"   "+_global.bin_a_hex(array_ins[ins_act+1]);

if (andop2reg == 2) {
	r2_ins_and1.setTextFormat(format_vermell);
} else if (andop2reg == 3) {
	r3_ins_and1.setTextFormat(format_vermell);
} else if (andop2reg == 4) {
	r4_ins_and1.setTextFormat(format_vermell);
} else if (andop2reg == 5) {
	r5_ins_and1.setTextFormat(format_vermell);
} else if (andop2reg == 6) {
	r6_ins_and1.setTextFormat(format_vermell);
} else if (andop2reg == 7) {
	r7_ins_and1.setTextFormat(format_vermell);
}
```

Relatives a l'execució de la instrucció (2a fase)

```
stop();
var andop1reg:Number;
var andop2reg:Number;
var andop1:String = next_ins.substr(2, 3);
var andop2:String = next_ins.substr(5, 3);
if (andop1.substr(0, 1) == "0") {
	var op1_temp_str:String = "1"+andop1;
	var op1_temp_dec:Number = parseInt(op1_temp_str, 2);
	andop1reg = op1_temp_dec-8;
} else {
	andop1reg = parseInt(andop1, 2);
}
if (andop2.substr(0, 1) == "0") {
	var op2_temp_str:String = "1"+andop2;
	var op2_temp_dec:Number = parseInt(op2_temp_str, 2);
	andop2reg = op2_temp_dec-8;
} else {
	andop2reg = parseInt(andop2, 2);
}

and_ops.text = "AND R"+andop1reg+" "+"R"+andop2reg;

var val_temp_dec_op1:Number = parseInt(Registres[andop1reg], 2);
var val_temp_dec_op2:Number = parseInt(Registres[andop2reg], 2);
val_temp_dec_op1 &= val_temp_dec_op2;

var resultat_final_and:String = val_temp_dec_op1.toString(2);
var long_resultat = resultat_final_and.length;

for (var i:Number = long_resultat; i<8; i++) {
	resultat_final_and = "0"+resultat_final_and;
}

//evitem escriptura en R0 i R1
if ((andop1reg != 0) && (andop1reg != 1)) {
	Registres[andop1reg] = resultat_final_and;
}

//mirem si l'operació ha donat 0
if (resultat_final_and == "00000000") {
	Z = 1;
} else {
	Z = 0;
}
text_valor_Z.text = "Z = "+Z;

textand2.text = "AND R"+andop1reg+"="+Registres[andop1reg]+";Z="+Z;

rb_ins_and2.text = "RB="+_global.bin_a_hex(RegistreB);
r2_ins_and2.text = "R2  "+_global.bin_a_hex(Registres[2]);
r3_ins_and2.text = "R3  "+_global.bin_a_hex(Registres[3]);
r4_ins_and2.text = "R4  "+_global.bin_a_hex(Registres[4]);
r5_ins_and2.text = "R5  "+_global.bin_a_hex(Registres[5]);
r6_ins_and2.text = "R6  "+_global.bin_a_hex(Registres[6]);
r7_ins_and2.text = "R7  "+_global.bin_a_hex(Registres[7]);
mem0_ins_and2.text = (ins_act-1) +"   "+_global.bin_a_hex(array_ins[ins_act-1]);
mem1_ins_and2.text = (ins_act) +"   "+_global.bin_a_hex(array_ins[ins_act]);
mem2_ins_and2.text = (ins_act+1) +"   "+_global.bin_a_hex(array_ins[ins_act+1]);
ins_act = ins_act+1;

pc_ins_and2.text = "PC="+ins_act+"d";
if (andop1reg == 2) {
	r2_ins_and2.setTextFormat(format_vermell);
} else if (andop1reg == 3) {
	r3_ins_and2.setTextFormat(format_vermell);
} else if (andop1reg == 4) {
	r4_ins_and2.setTextFormat(format_vermell);
} else if (andop1reg == 5) {
	r5_ins_and2.setTextFormat(format_vermell);
} else if (andop1reg == 6) {
	r6_ins_and2.setTextFormat(format_vermell);
} else if (andop1reg == 7) {
	r7_ins_and2.setTextFormat(format_vermell);
}
```

Com a comentari per aquests dos últims apartats de l'annex, val a dir que les operacions d'una sola fase comparteixen les característiques comuns a les de dues fases (com la part on es realitza l'operació) i difereixen a les parts en les quals es fa servir, per exemple, el RegistreB o a les parts o es pot arribar a fer el salt condicional i donar-li format als camps de text.

RESUM / RESUMEN / ABSTRACT

Resum

Aquest Projecte pretén crear un simulador d'una màquina algorísmica, de caràcter didàctic amb Adobe Flash CS3 per a reproduir-lo amb Adobe Flash Lite Player, que és la versió per a dispositius mòbils.

Simularà el comportament de la màquina algorísmica anomenada FEMTOPROC, que és capaç d'interpretar 4 instruccions molt senzilles: ADD, AND, NOT i JZ (jump if zero).

Les diferents instruccions introduïdes que compondran un programa seran emmagatzemades en una memòria de 64 posicions de 8 bits cadascuna i hi haurà un Banc de Registres amb 8 registres de 8 bits, que es podrà inicialitzar al començament de la simulació.

Resumen

Este Proyecto pretende crear un simulador de una máquina algorítmica de carácter didáctico con Adobe Flash CS3 para reproducirlo con Adobe Flash Lite Player, que es la versión para dispositivos móviles.

Simulará el comportamiento de la máquina algorítmica llamada FEMTOPROC, que es capaz de interpretar 4 instrucciones muy sencillas: ADD, AND, NOT y JZ (jump if zero).

Las diferentes instrucciones introducidas que compondrán un programa serán almacenadas en una memoria de 64 posiciones de 8 bits cada una y habrá un Banco de Registros con 8 registros de 8 bits, que se podrá inicializar al principio de la simulación.

Abstract

This project aims to create a simulator of an algorithmic machine with didactic purposes with Adobe Flash CS3, in order to reproduce it with Adobe Flash Lite Player, which is the version for mobile devices.

It will simulate the behavior of the algorithmic machine called FEMTOPROC, which is capable of interpreting 4 very simple instructions: ADD, AND, NOT and JZ (jump if zero).

The different instructions that will compose a program will be stored in a 64x8bits memory and there will be a Bank of Records with 8 records of 8 bits, which will be initialized at the beginning of the simulation.